# Executable Specifications: End-to-End Acceptance Testing with SpecFlow

**Module 4 Setup Instructions**

In Module 4 we implement end-to-end acceptance test for the new REST API in the solution, using SpecFlow with the HttpClient for end-to-end testing.

## Sixeyed.SpecFlowItAll

This is the demo solution for the course. In this module we walk through the code and show the features we will be writing acceptance tests for.

### Pre-requisites

Follow the setup instructions from Module 3.

You can run the solution and the tests without SpecFlow installed, but for the editor experience you should install SpecFlow 1.9 from the Visual Studio Extension gallery: http://visualstudiogallery.msdn.microsoft.com/9915524d-7fb0-43c3-bb3c-a8a14fbd40ee

The automated test uses Firefox, so you will need a version installed, or you will have to modify the SpecFlow steps to use a different WebDriver (e.g. for Chrome or IE).

You will need to have MSMQ installed if you want to run the message handler tests. Instructions vary for different versions of Windows – see:

http://msdn.microsoft.com/en-us/library/aa967729%28v=vs.110%29.aspx

### loader.io

If you want to run your own load tests, you will need deploy your REST API to a public Web address, and sign up for an account with loader.io. Replace the loader and performance test config values in the app.config for the acceptance test project:

* api.url.perf
  + the URL for your cloud deployment of the REST API
  + e.g. http://sc-specflow-api.cloudapp.net
* loader.url
  + the URL for your load test defined in loader.io
  + e.g. https://api.loader.io/v2/tests/6d27cbe3405840e2fe457e2ac6c70b2e/run
* loader.key
  + the API key for your loader.io account
  + e.g. 3f199b63f1e794b062f2e67243348d63

### Running the Tests

Build the solution and in Visual Studio Test Explorer you will see the GetProduct SLA test listed. Run that test and if everything is configured and deployed correctly, after 30 seconds the test should go green validating that your cloud deployment can support 500 requests/second.